





Génie des Systèmes Interactifs

Philippe Palanque

palanque@irit.fr - http://liihs.irit.fr/palanque

Logiciels Interactifs et Interaction Homme-Systèmes

Université Paul Sabatier (Toulouse 3)



Plan du cours

- Introduction et définitions
- Processus de développement des SI
- Les techniques de spécification formelle pour les SI
- Les techniques d'ingénierie des SI
- Les méthodes de conception de SI
- Perspectives de recherche

3

Introduction à l'IHM

4

Définition des SI

5

Cycle Cascade Court

Requirements analysis and definition

System and software design

Implementation and unit testing

Integration and system testing

6

Cycle de développement en cascade

Requirements specification

Architectural design

Detailed design

Coding and unit testing

Integration and testing

Operation and maintenance

7

Cycle de développement en V

Analyse du besoin

Spécification

Conception Préliminaire

Conception détaillée

Codage

Tests Unitaires

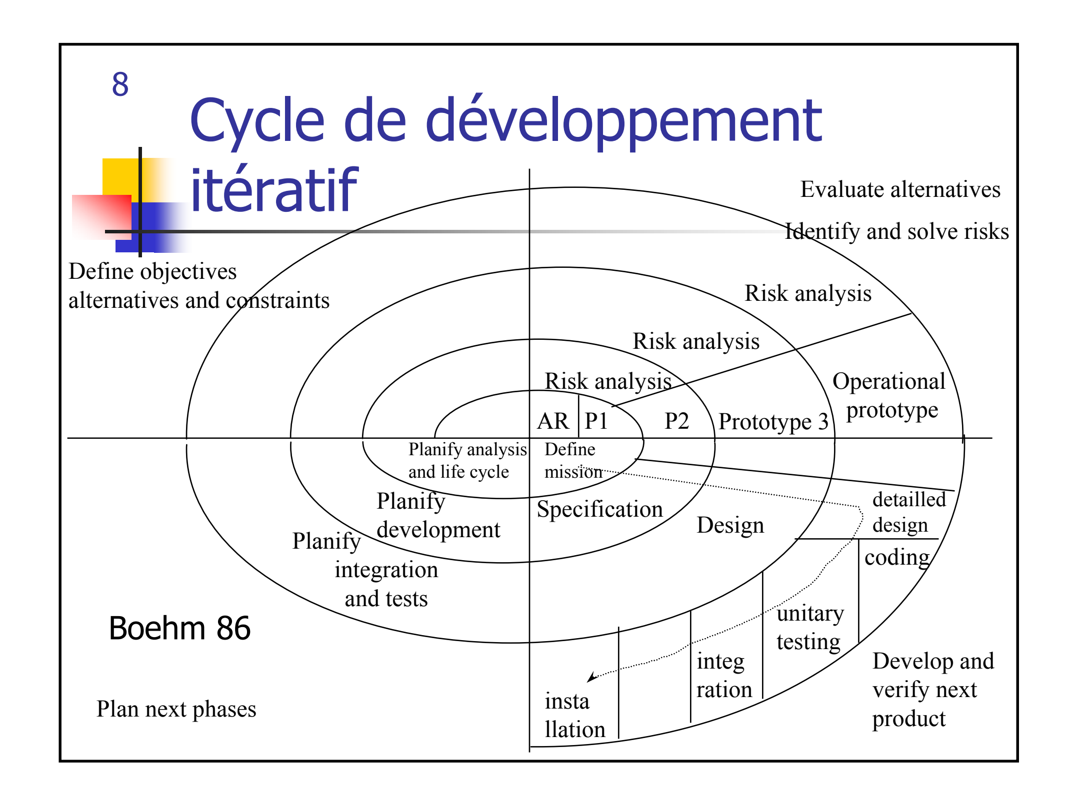
Intégration

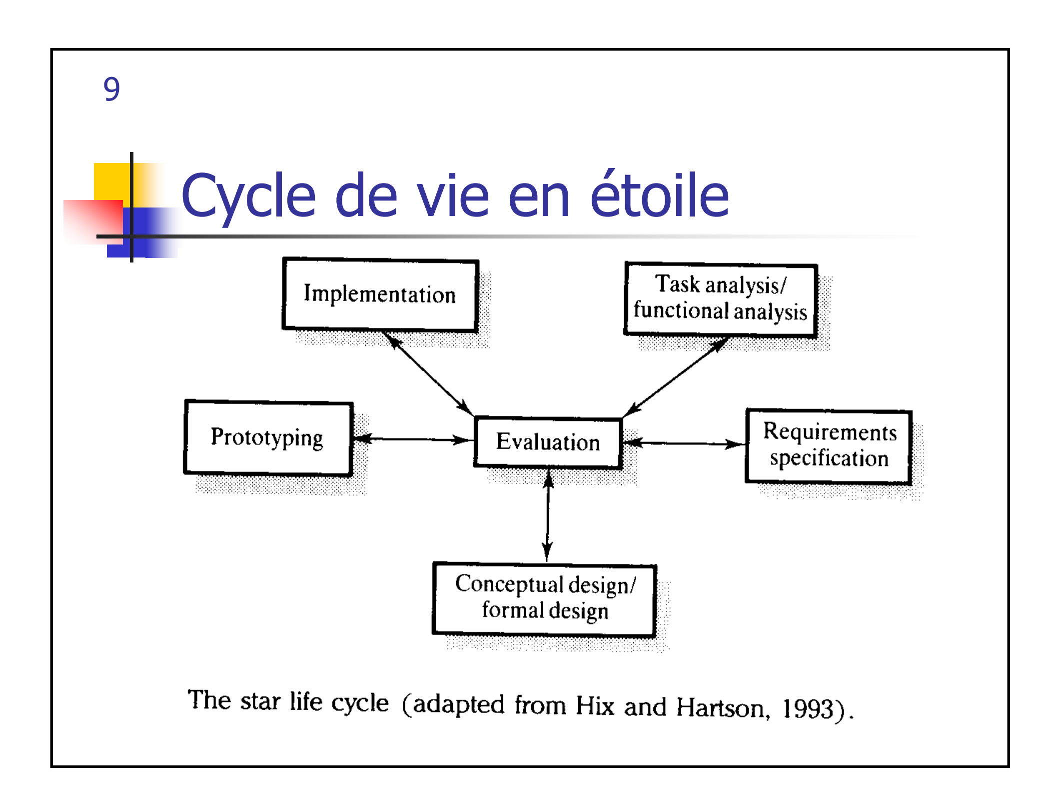
Validation

Déploiement

8

Cycle de développement itératif





The star life cycle (adapted from Hix and Hartson, 1993).

10



Méthodologie de Conception

Interfaces Homme Machines :
Domaine pluri-disciplinaire
"Science" non exacte :

La conception d'une interface

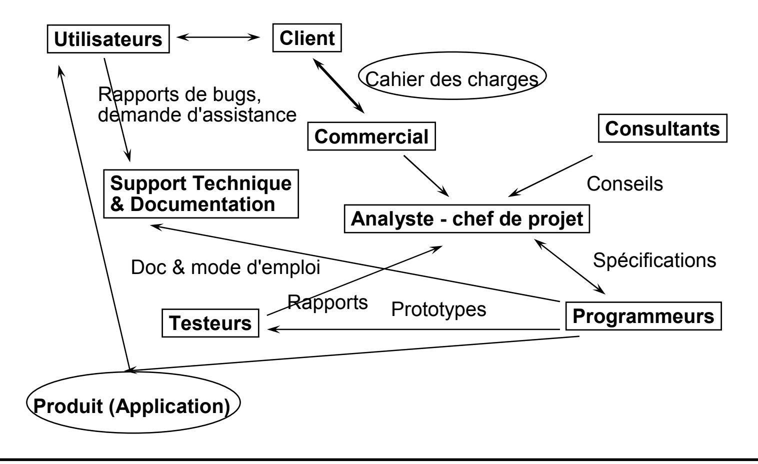
Relève plus du **savoir-faire** que d'une méthodologie précise

=> Méthodologie appropriée de conception, différente de celles des applications classiques



Conception dans la pratique

Circuits de l'information beaucoup plus complexes





Points faibles de l'organisation

• Trop d'intervenants

Les acteurs principaux (analyste<-> utilisateurs) sont les personnes les plus éloignées du schéma

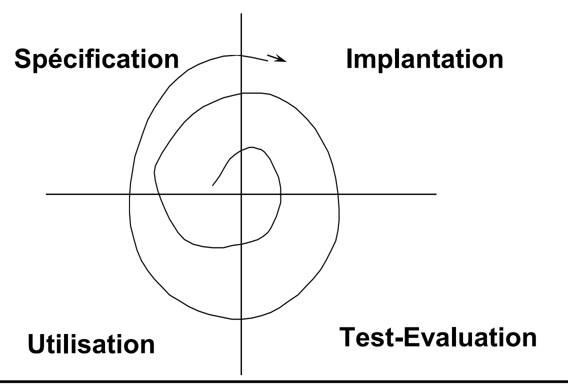
=> essayer de réduire les circuits de l'information

• Les personnes les plus qualifiées ont peu droit à la parole et aux choix décisifs



Les approches itératives

Évolution "En spirale" vers le produit final





Approches "Prototype"

Prototype "Haute Fidélité" :
-> réduire le nombre de boucles pour aller au plus vite vers l'application finale

Utilisation des outils interactifs de développement.

Inconvénients :
- lenteur des boucles d'itération
- difficultés à dégager clairement à chaque étape les problèmes essentiels:
-> la réalisation du prototype fait intervenir presque en même temps les 4 phases
- Les outils de prototypages imposent leur limites, parfois assez importantes

15



Prototypage - Maquetage

Prototype : diffère du produit final par le processus de conception

Maquette : diffère du produit par l'échelle (taille, nombre de fonctionnalités, …)

En IHM le sens de ces mots a été altéré

Prototype : produit qui fonctionne (des parties de chacune des couches du modèle de Seeheim ont été développées)

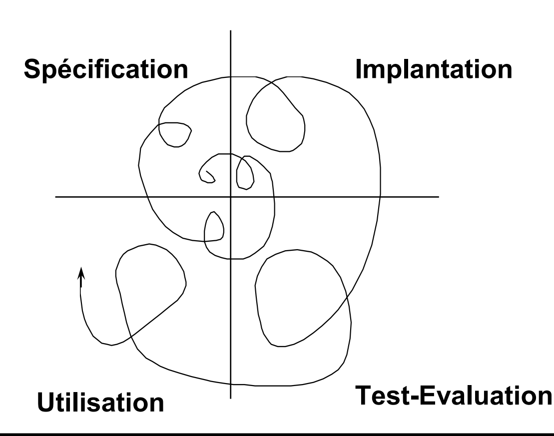
Maquette : l'ensemble de la partie présentation a été réalisée mais les fonctionnalités ne sont pas mises en œuvre (on voit la statique de l'interface mais pas la dynamique)

16



Approches "super-itératives"

Réaliser des itérations à chaque étape du processus:



17

Prototypage Jetable

Preliminary requirements
Build prototype
Evaluate prototype
adequate?
no
yes
Final requirements

18

Prototypage Incrémental

Identify components
Req
Arch
Det
Impl
Int
system complete?
no
deliver increment
yes
dèliver system
Operation and maintenance

19



Spécifications itérées

Approche "Basse Fidélité" :
Partir de schémas bruts et très simples qui exposent
uniquement les problèmes importants puis raffinements progressifs
Dessins et maquettes "manuelles" suggérant le mode de fonctionnement
sans rentrer dans les détails de l'interface, distrayants.

+ efficace et + abstrait qu'un prototype

Interaction constante avec les futurs utilisateurs

Faire intervenir à ce moment là seulement les consultants externes:
graphistes, ergonomes, spécialistes du domaine…

Envisager des **Cours de dessin, d'expression graphique.**

20



Prototypage-Réalisation

• **Limiter au maximum le nombre des intervenants:**
Lorsque les spécifications sont assez claires,
toute intervention extérieure ne peut que faire perdre du temps:
- de personnes => moins d'incohérences dans le produit
- 1 personne motivée travaille mieux que 10 ensemble:
tous les grands progiciels sont écrit par une équipe d'au plus 4 personnes

• **Ne pas hésiter à revenir en arrière, voire à zéro dès le début:**
prototypes "jetables" :

malgré les problèmes de délais, il est souvent plus rapide de repartir
de rien en tirant profit de l'expérience acquise, que de traîner
des problèmes de conception originaux lors de plusieurs itérations.

Au bout d'une certaine étape, il est impossible de revenir en arrière
=> se prémunir au maximum des échecs dû à des erreurs faites
au début.



Coopération avec l'utilisateur

Techniques de "Collaborative Design":
Au départ, utilisées en Scandinavie pour faire accepter l'informatisation aux syndicats des grandes entreprises.

Se sont avérées très profitables.

-> Les seuls vrais connaisseurs du système, de ses limites et de ses capacités sont :
- les programmeurs (+ encore que les concepteurs)
- les utilisateurs

Technique:
Les futurs utilisateurs et les concepteurs partagent un plan de travail. Les concepteurs exposent leur projet, les utilisateurs annotent le projet, avec de petites notes, chacune relatant un point précis qui semble poser problème.



Mise en œuvre

• Création d'un modèle conceptuel:

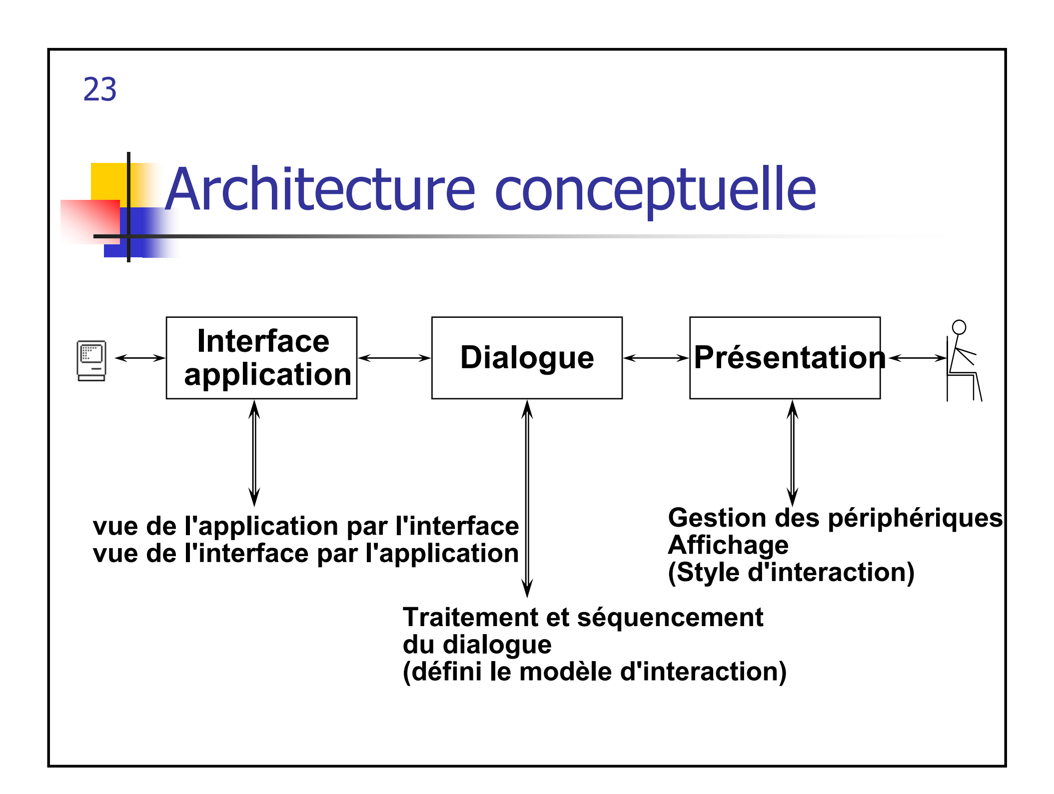
- définir les objets utilisés
- les opérations à effectuer
- les concepts et les enchaînements logiques

• Représentation du Contrôle

• Choix et Spécification du Modèle d'Interaction

-> utiliser les dessins et schémas sur papier pour
- la représentation des données
- la formulation des actions

-> penser aux règles de cohérence et concision.





Conception des IHM

Il faut concevoir les trois parties du modèle de Seeheim:

La présentation : ce que l'utilisateur voit de l'application

Le dialogue :

- qu'est-ce que l'utilisateur a la possibilité de faire
- comment l'utilisateur agit sur la présentation
- l'influence de son action sur ce qu'il pourra faire ensuite

Le noyau fonctionnel :

- les fonctions réalisées par l'application
- les données manipulées par l'application

25



Exercice : Interface de Voiture

Faire l'interface de d'utilisation d'une voiture en simulant son utilisation par un ordinateur. Il faudra représenter graphiquement les deux principales tâches:

- le pilotage
  - afficher les compteurs
  - actions offertes au pilote
  - donner du retour d'information
- la navigation : on peut avoir besoin d'informations du genre
  - proposer une route en fonction de la durée de voyage
  - proposer une route en fonction de son taux de chargement
  - proposer une route en fonction du départ et de la destination
  - communication entre véhicules
- on suppose que la manipulation du volant, pédales et le levier de vitesse restent telles que offertes dans les véhicules actuels

Faites cet exercice par binôme. Chacun des éléments du binôme est expert soit dans la tâche de pilotage soit dans la tâche de navigation.

26



Exercice : Aéroport

- Vous êtes dans un avion en train d'atterrir
- Vous avez une correspondance très courte
- Vous avez faim
- Déterminez un système de distribution de nourriture permettant d'obtenir la nourriture le + rapidement possible

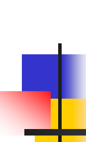
27

Modèle de Seeheim 83

Utilisateur ↔ Composant Presentation ↔ Contrôleur du Dialogue ↔ Interface du noyau fonctionnel ↔ Noyau fonctionnel

28

Modèle Architectural



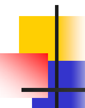
Description Formelle de Systèmes Interactifs



Références (livres)

- Formal Methods in HCI, Harrison & Thimbleby 90, Cambridge University Press
- Formal Methods for Interactive Systems, Dix 91, Cambridge University Press
- Formal Methods in HCI, Palanque & Paterno 97, Springer Verlag

31



Références (conférences)

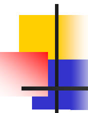
- Eurographics workshops DSV-IS (Design, Specification and Verification of Interactive Systems) depuis 94 publiés par Springer Verlag (LNCS depuis 2000)
- CHI 96 workshop on Formal Methods and HCI
- CHI 98 workshop on designing user interfaces for safety critical systems
- Mini-track de FM 99
- SUCA 2000 workshop (Safety and Usability Concerns in Aeronautics)

32



Références (journaux)

- Pas de revue dédiée
- Des "special issues"
  - Journal of Visual Language and Computing (vol 10, n°3, 1999)
  - ACM Transactions on Computer Human Interaction (début 2000)
  - Interacting with computers (BCS HCI group journal)

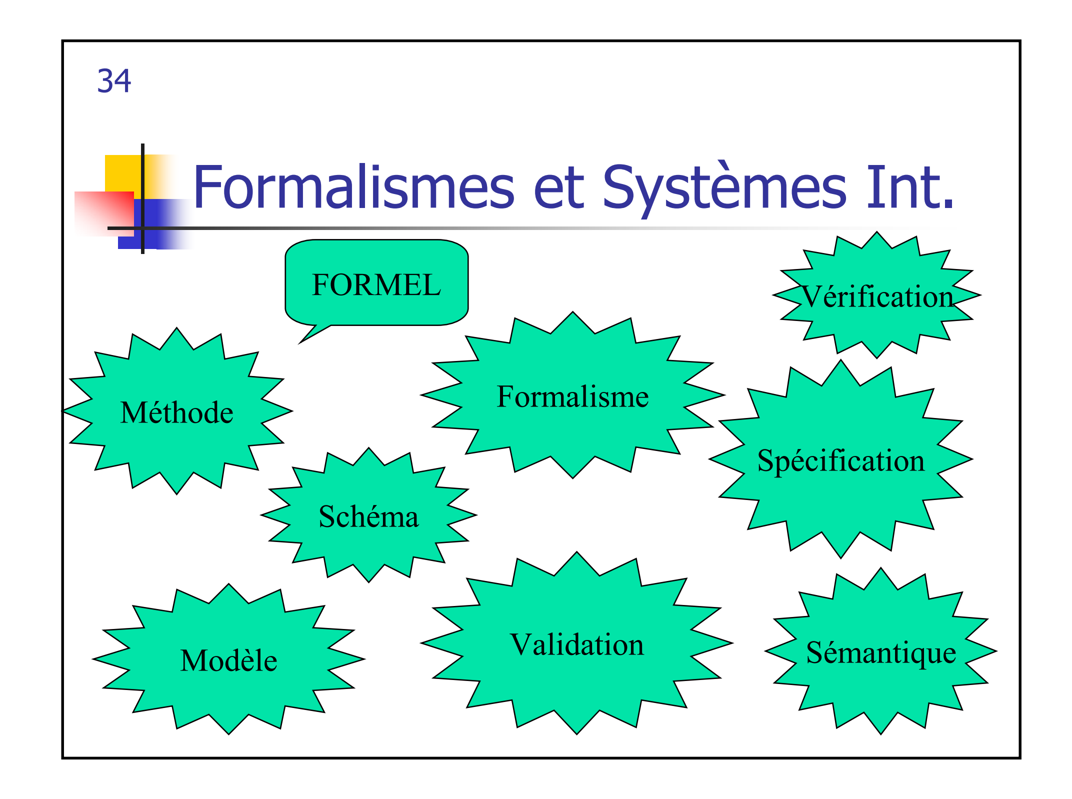


Groupes de travail

- Pas de groupe dédiés
- Groupe IFIP 2.7 (13.4)
- Groupe FLASHI du GDR-PRC-CHM (de 96 à 98)
- Groupe ALF du GDR-PRC-I3 depuis 98

- Marginalement
  - Groupe IFIP 13.5 Human Error
  - Groupe IFIP 13.2 Methodologies

Formalismes et Systèmes Int.



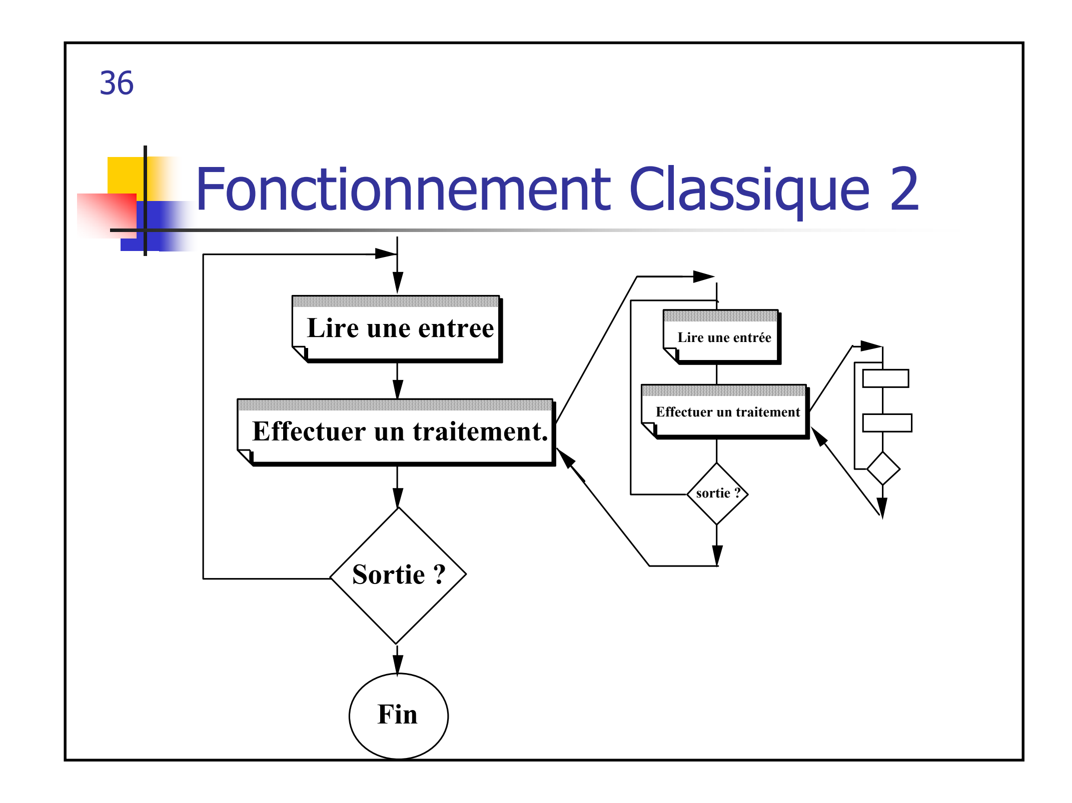


Fonctionnement Classique

```
Début
choix = '1';
Tantque choix <> '9' faire
    affiche-menu;
    lire(choix);
    case choix of
    1 : ajouter;
    2 : modifier;
    3 : supprimer;
    9 : Quitter;
    Fin Case
FintTantque
Fin
```

```
Procédure Ajouter;
début
rep = 'o';
Tantque rep <> 'n';
    dessin-écran;
    lire(nom);
    lire(prenom);
    ...
    écrire('voulez-vous
    continuer ?');
    lire(rep)
FinTantque
Fin
```


Fonctionnement Classique 2



37

Fonctionnement Interne

Control flow

Data flow

Event Queue

Read event

INIT

Dispatch event

Event-handler 1

Event-handler 2

. . . . . .

. . . . . .

Event-handler ..

Event-handler n

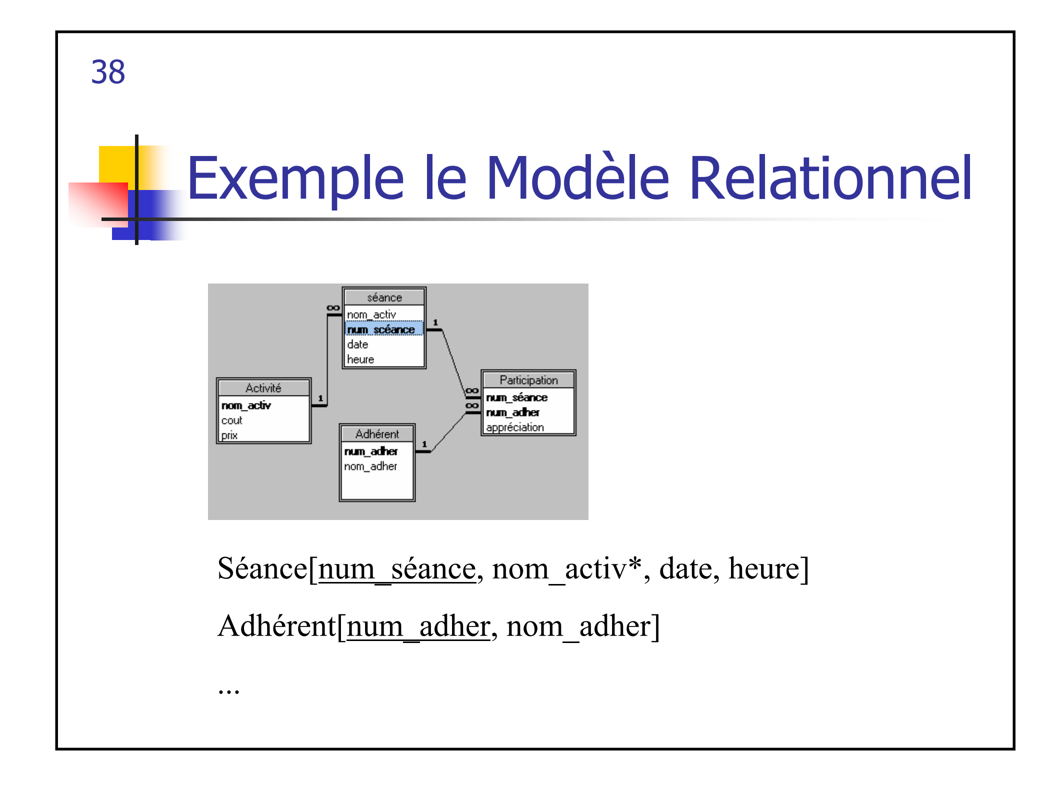
Dialogue state

No

completed

Yes

END



38

Exemple le Modèle Relationnel

Séance[num_séance, nom_activ*, date, heure]

Adhérent[num_adher, nom_adher]

...

39

Exemple le Modèle E/A

Personnes
Nom
Prénom
Email
Web

Examinateur

Auteur

Etablissement
NuméroEtab
NomEtablissement
Adresse
Adresse2
BoitePostale
CodePostal
Ville
Cedex
NumCedex

Directeur

Rapporteur

Thèse
Numéro
Titre
Résumé
DateSoutenance
Mention
Contenu

Appartient

Porte

Discipline
NuméroDomaine
NomDeDomaine

0,n

40

Processus de conception "Classique"

Requirements Informels

Manuel

Spécification — *Spec 1* … *Spec n*

Conception — *Design 1* … *Design n*

Développement — *Prog. 1* … *Prog. n* ⟷ *Tests Utilisateurs*

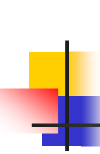
41



Qu'est-ce qu'un "Modèle"

- Concepts et liens entre concepts
- Exemple : le Modèle Entité/Association
  - Concepts : Classe d'Entité, Classe d'Association, Attribut, Domaine, Identifiant....
  - Liens entre concepts : "Une CA est un sous-ensemble du produit cartésien de 2 CE"

42



Que doit-on attendre d'un "Modèle"

- Complétude (On peut exprimer tout ce que l'on souhaite)
- Consistance (On ne peut pas exprimer d'énoncés contradictoires)
- Généralité (Indépendant d'un domaine d'application particulier... Toujours ?)

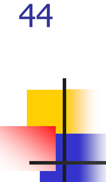
43



Qu'est-ce qu'un "Formalisme"?

- Conventions de représentation des concepts d'un Modèle
  - Lexique (graphique ou textuel)
  - Syntaxe concrète (séparateurs, terminaux, ...)

44



Que doit-on attendre d'un "Formalisme" ?

- Expressivité
- Concision
- Complétude par rapport au Modèle (contre-ex : le formalisme Entité Association)
- Proximité accrue de la représentation avec un domaine d'application
- "the only difficult problem the author solved was understanding his own notation"L.Lamport

45



Bug : Avec un "Formalisme" on fait des "modèles"

- modèle : représentation idéale d'une situation du monde réel, limitée aux concepts couverts par un Modèle.
- Objectif : analyser le modèle pour en tirer des enseignements sur la situation du monde réel
- Méta-modèle : …

46



Pourquoi modéliser un SI ?

- Une description abstraite du système
  - indépendante de l'implémentation
  - qui n'intègre pas trop tôt les détails
- Décrire les sorties à produire en fonction des entrées
- Permettre la discussion entre les différents intervenants (à un instant donnée et tout au long du processus de développement)
  - garder le résultat des discussions
  - garder la trace des discussions ???

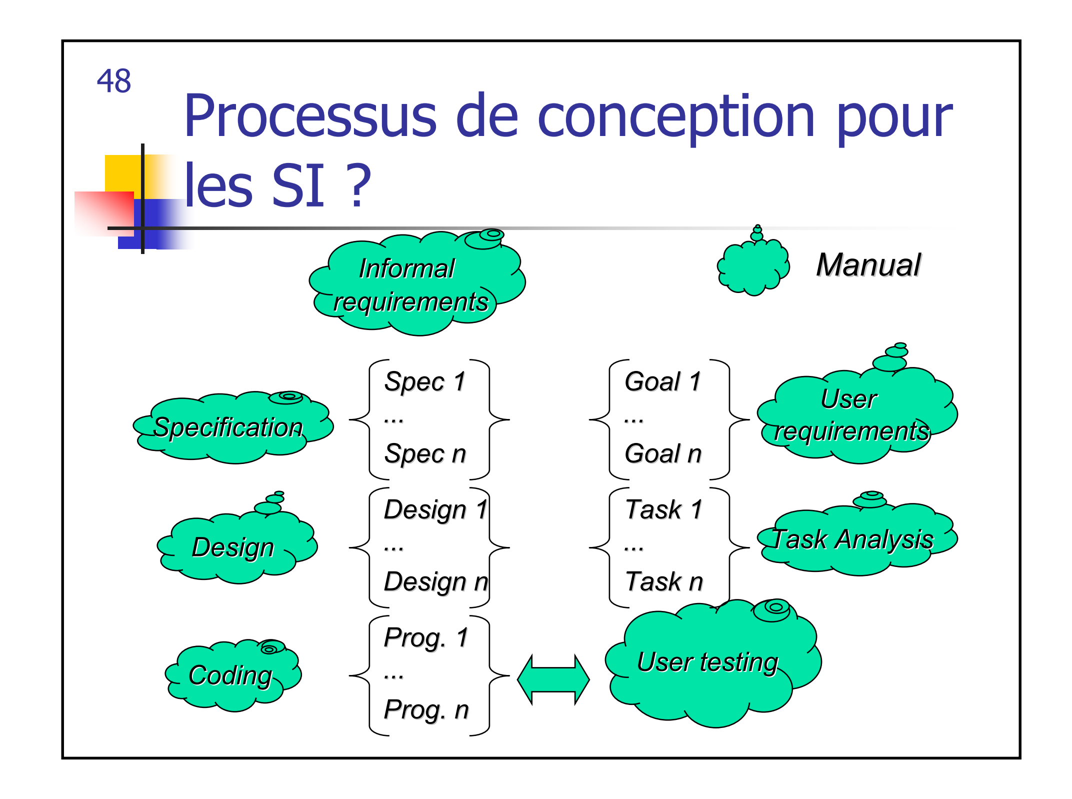
47

Qu'est-ce qu'un modèle du système pour SI ?

- Décrit à la fois données et actions du système
- Décrit le comportement du système
  - Quelles sont les actions offertes
  - Quand une action est disponible (en fonction de l'état du système)
  - Quel est l'effet d'une action sur l'état du système
- Décrit l'aspect externe
  - comment le système est rendu perceptible à l'utilisateur
  - comment l'utilisateur peut interagir avec le système

48

Processus de conception pour les SI ?



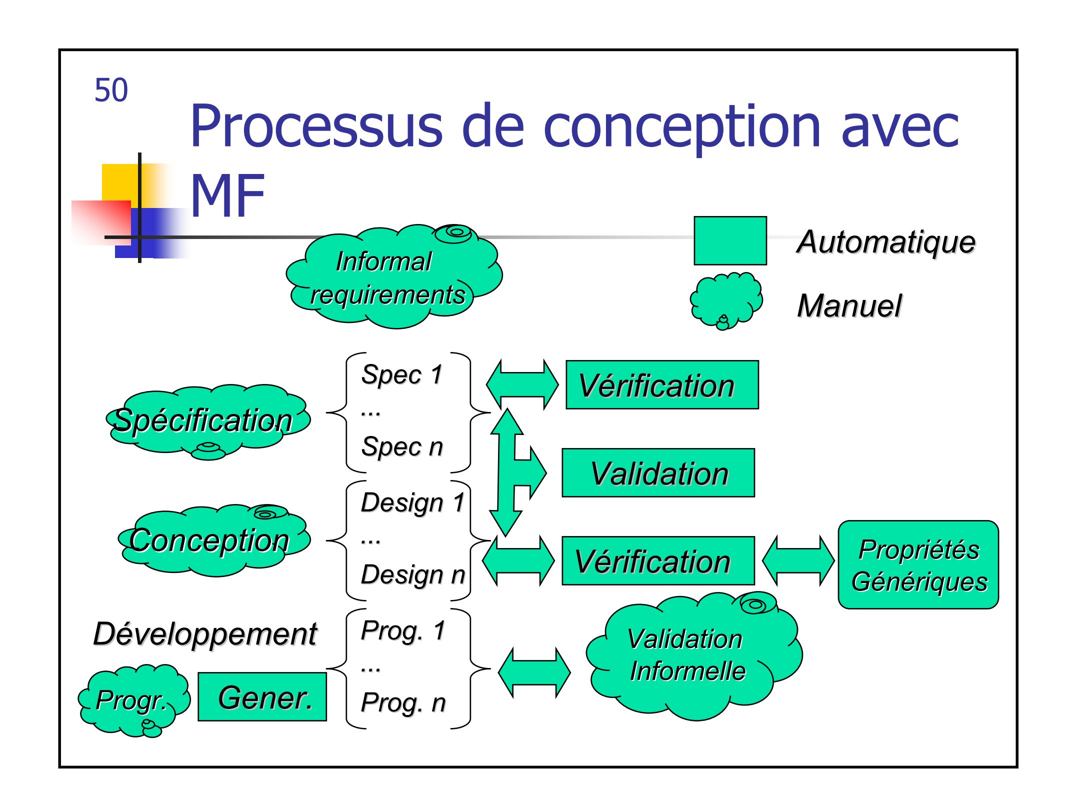
49

Pourquoi concevoir les SI formellement ?

- Pour gérer la complexité
- Pour limiter l'intervention humaine en observation (contrôler les modèles)
- Pour limiter l'intervention humaine en traduction (écriture du code)
- Pour raisonner
  - en validation
  - en vérification
- Pour atteindre trois buts fondamentaux
  - fiabilité : propriétés spécifiques et génériques
  - efficacité : performances du système, de l'utilisateur (charge de travail) et du couple utilisateur/système (au niveau des tâches)
  - utilisabilité

50

Processus de conception avec MF



Processus de construction des modèles

Preliminary model

Formal analysis

Check Desired Properties

Ok

Not Ok

Mending of the model

Model, i^{th} iteration

Relations entre tâches et système

Maintain task and system models consistency

Preliminary system model

Formal system modelling

i^{th} iteration

Formal task modelling

i^{th} iteration

Preliminary task model

Quantitative analysis

Check Objectives

Proposals for improving the system model

Towards Usability Testing

53

Requirements modelling
i th iteration

Models
Activities

Qualitative analysis
Maintain conformance to requirements
Maintain task and system models consistency

System modelling
i th iteration

Task modelling
i th iteration

Quantitative analysis
Complexity evaluation
Performance evaluation

Improve the system model
Not Ok
Check Objectives
Ok
Towards Usability Testing

54



Exemples de modèles (requirements)

- Un formalisme déclaratif est nécessaire
- Requirements
  - Toute clearance envoyée par un contrôleur à un avion p est reçue par cet avion

    ∀ p Planes, ∀ req DLRequest,

    AG[send(req,p)]AF<receive(req,p)>true
  - Une clearance data-link envoyée par un contrôleur à un avion p sera uniquement reçue par cet avion

    ∀ p,p' Planes, ∀ req DLRequest,

    AG[send(req,p)]AG[not(receive(req,p')]true

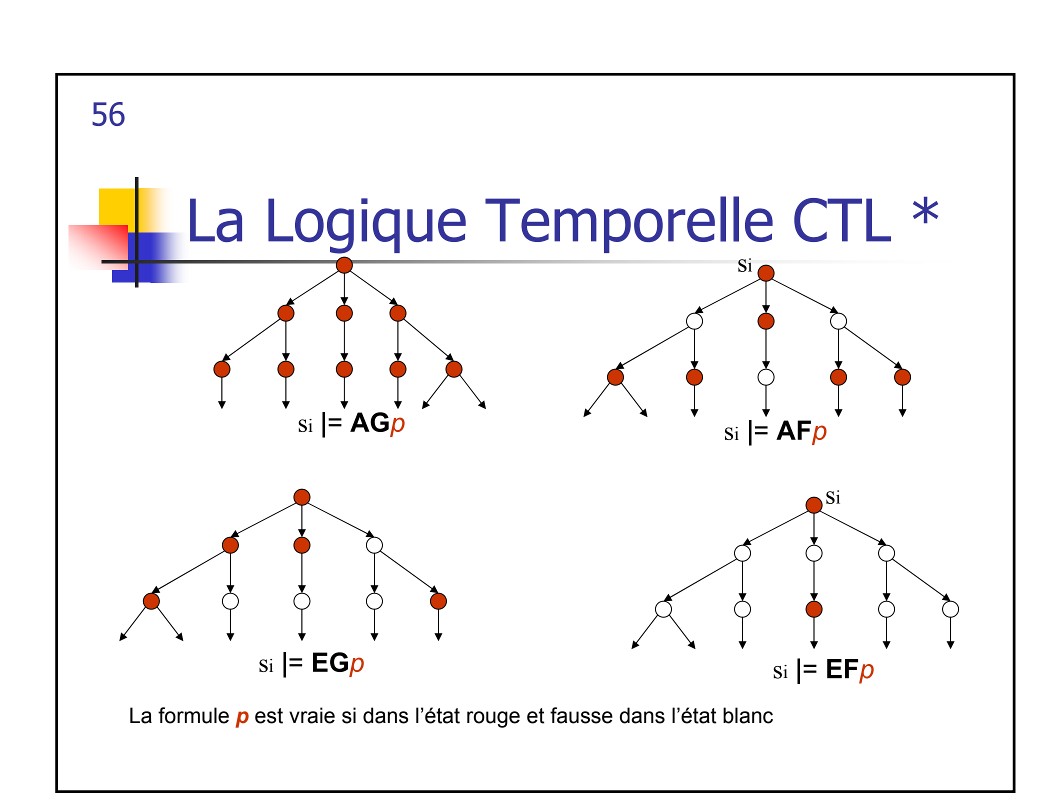
55

La Logique Temporelle CTL * (Computational Tree Logic Star)

- Un état a un ou plusieurs successeurs
- L'ensemble des états représente un arbre infini
- Opérateurs:
  - **A** (tous les futurs possibles); **E** (un futur possible) + {**F inévitablement**, **G toujours**, **X suivant**, **U jusqu'à**}
- Connecteurs logiques:
  - $\wedge$ (et); $\vee$ (ou); ˥ (non); $\Rightarrow$ (implication)

56

La Logique Temporelle CTL *



La formule ***p*** est vraie si dans l'état rouge et fausse dans l'état blanc

La Logique Temporelle CTL *

Si

si |= AX*p*

Si

si |= EX*p*

Si

si |= A(*p* U *q*)

Si

si |= E(*p* U *q*)

Vérification de la compatibilité des modèles

- Tous les objets du modèle de tâches existent dans le modèle de données du système
- Toutes les actions du modèle de tâche sont offertes par le système
- Toutes les actions du système existent dans le modèle de tâche
- Toutes les séquences d'action du modèle de tâche sont "licites" dans le modèle du système
- Toutes les contraintes (requirements) exprimés sont vrai sur le modèle du système et/ou sur le modèle de tâche

59



Fondements des approches formelles pour les SI

- décrire à la fois les états et les événements
- décrire à la fois la structure de donnée et la structure de contrôle
- offrir des mécanismes de structuration
- décrire la concurrence
- décrire les aspects temporels
- faire tout cela de façon formelle

60



Aspects états et événements

- Systèmes réactifs
- Dirigé par événement
- Code difficile à gérer sans représentation exploitable des états
- Approches
  - approches venant de la conception des SR
  - langages réactifs ou synchrones (esterel, Lotos)
  - utilisation méthodologique des RdP

61



Aspect structure de donnée / structure de contrôle

- La crise du logiciel à montré les limites de la séparation données/traitements (maintenabilité, évolutivité, réutilisabilité, ...)
- Approches
  - mélanger deux approches (CSP-Z, Object-Z, Full LOTOS, …)
  - intégrer deux approches (ICOs)

62



Mécanismes de structuration

- Gestion de composants complexes
- Compréhensibilité des modèles
- Réutilisabilité
- Evolutivité
- Approches
  - composition (agrégation, association, …)
  - communication (client-server, actors)
  - macros and plugs

63



Concurrence

- Concurrence avec les dispositifs d'entrée et de sortie
- Collecticiels
- Dialogues multi-fils
- Approches
  - formalismes textuels (CSP, CCS, LOTOS, Logiques Temporelles)
  - formalismes graphique (réseaux de Petri, Statecharts)

64



Aspects temporels

- Systèmes multimodaux
- Animation (évolution basée sur le temps)
- Alarmes
- Événements calendaires

- Approches
  - procédurales (RdP, Lotos, ...)
  - déclaratives (Logiques temporelles, ...)

65



Caractère formel

- Plus facile de prouver que de tester
- Systèmes critiques
- Complétude, concision et non-ambiguïté
- Exécutabilité
- Approches
  - validation mathématique
  - génération de jeux de tests
  - génération automatique de code

66



Etat de l'art en FM et HCI

- Compréhension des SI
  - Qu'est-ce qu'un système réactif ?
  - Quelles sont les propriétés fondamentales des SI ?
  - Comment trouver et décrire les propriétés spécifiques à telle ou telle application interactive ?
- Ingénierie des SI
  - Quels sont les composants de base des SI
  - Comment décrire ces composants et leurs inter-relations ?
  - Comment ceci se situe par rapport au processus de développement ?

67



Compréhension des SI

- Architectures génériques pour les SI
  - PIE and red-PIE models (Dix 91)
  - Seeheim (Green 85) and Arch/Slinky (Bass 92)
- Propriétés génériques pour les SI
  - (Sufrin & He 90), (Dix 91)
  - Propriétés externes et internes (Gram & Cockton 96)
- A mi-chemin entre compréhension et ingénierie
  - CNUCE interactors (Paternò & Faconti 92)
  - York interactors (Duke & Harrison 93)

68



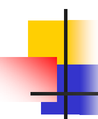
Ingénierie des SI

- Dialogue modelling WIMP
  - (Bastide & Palanque 90)
  - (Beck et al. 95)
- Model-Based UIMS (WIMP) (CADUI'96)
  - UIDE (Foley et al. 93)
  - Tadeus (Elwert & Schlungbaum 95)
  - PetShop (Bastide & Palanque 95)
- Design Patterns: PAC (Coutaz 87) & MVC



Propriétés génériques

- Propriétés génériques des systèmes
  - vivacité
  - sécurité
  - usure
- Propriétés génériques des systèmes interactifs
  - prédictibilité
  - observabilité
  - atteignabilité
  - Insistance



Les Automates Etendus

- Un automate étendu est un automate à états pouvant posséder :
  - des événements déclenchant des actions
  - des conditions de déclenchement des actions
  - des registres (variables numériques)
  - effectuer des actions sur les registres (affectation)
- Les événements, les conditions et les actions sont représentés
- sur les arcs sous la forme suivante :



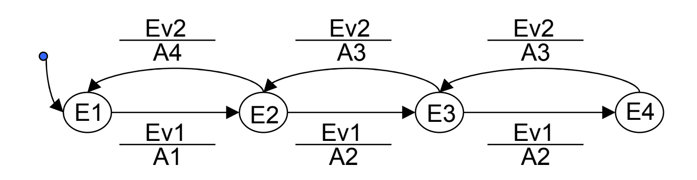
Les Automates Etendus (2)

Exemple : un système simple

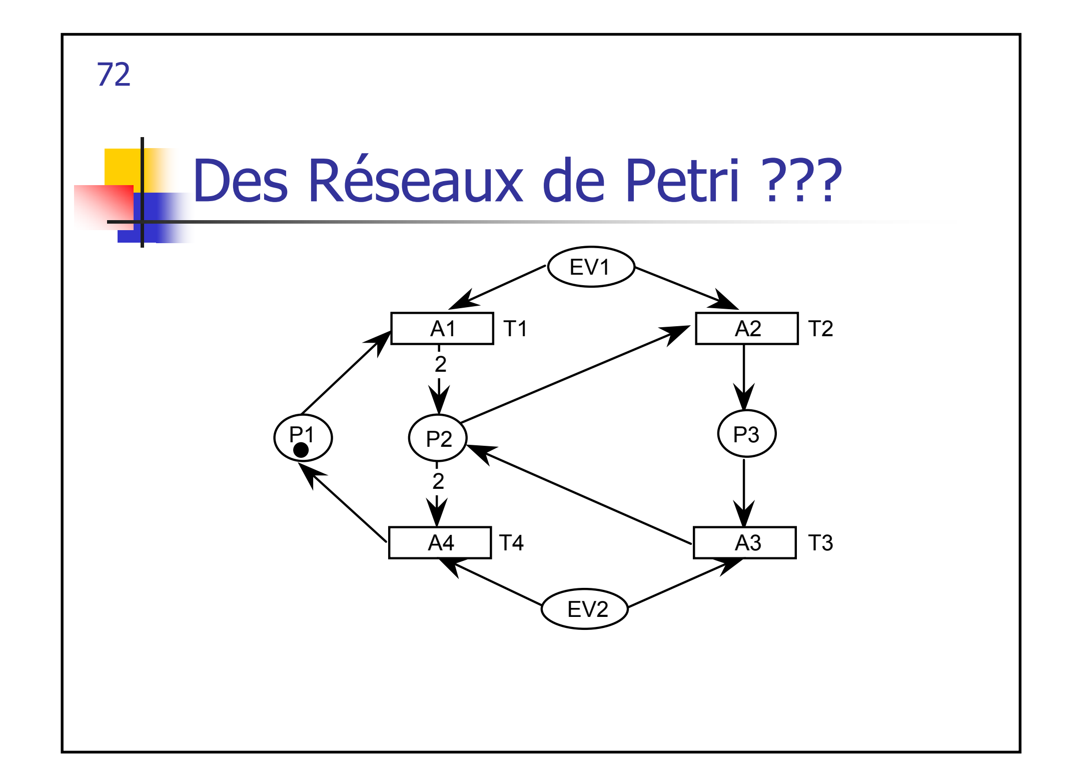
- **E = {E1, E2, E3, E4}, s0 = E1**
- **Ev = {Ev1, Ev2},**
- **A = {A1, A2, A3, A4},**

- **f : E x Ev -> A,**
  f(E1,Ev1) = A1,
  f(E2,Ev1) = f(E3,Ev1) = A2,
  f(E4,Ev2) = f(E3,Ev2) = A3,
  f(E2,Ev2) = A4

- **g : E x Ev -> E,**
  g(E1,Ev1) = g(E3,Ev2) = E2,
  g(E2,Ev1) = g(E4,Ev2) = E3,
  g(E3,Ev1) = E4,
  g(E2,Ev2) = E1.



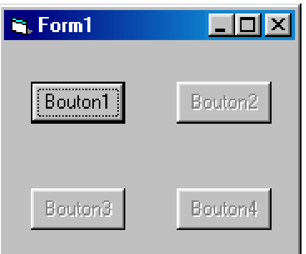
Des Réseaux de Petri ???

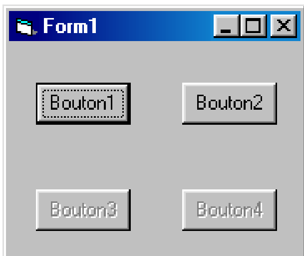




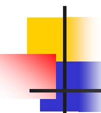
Exemple: les 4 boutons

- Spécification du comportement d'une application avec 4 boutons cycliques



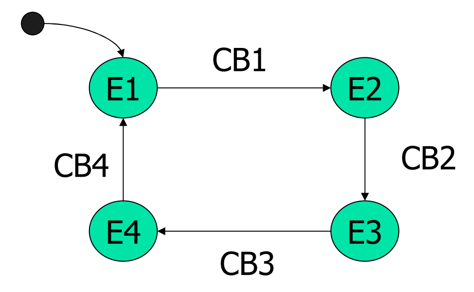


- Spécification du comportement d'une application avec 4 boutons alternatifs



Automate Exercice 1

- 4 événements CB1, CB2, CB3, CB4



Automate Exercice 2

- 4 événements CB1, CB2, CB3, CB4

Boucle d'affichage

- 2 boutons (start, stop)
- 1 compteur
- L'utilisateur doit pouvoir interrompre à tout instant

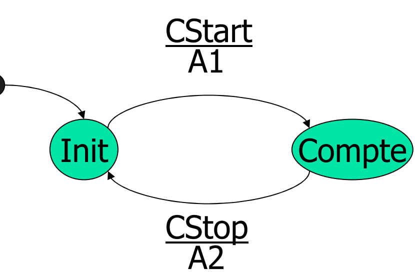

77



Automate Compteur

- A1:

```
For I=1 to 30000
    label.caption=I
Next I
```



- A2:
  Label.caption="Pouf"

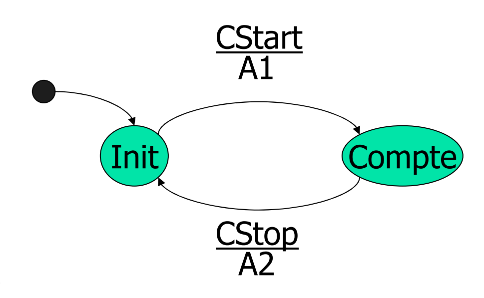
78



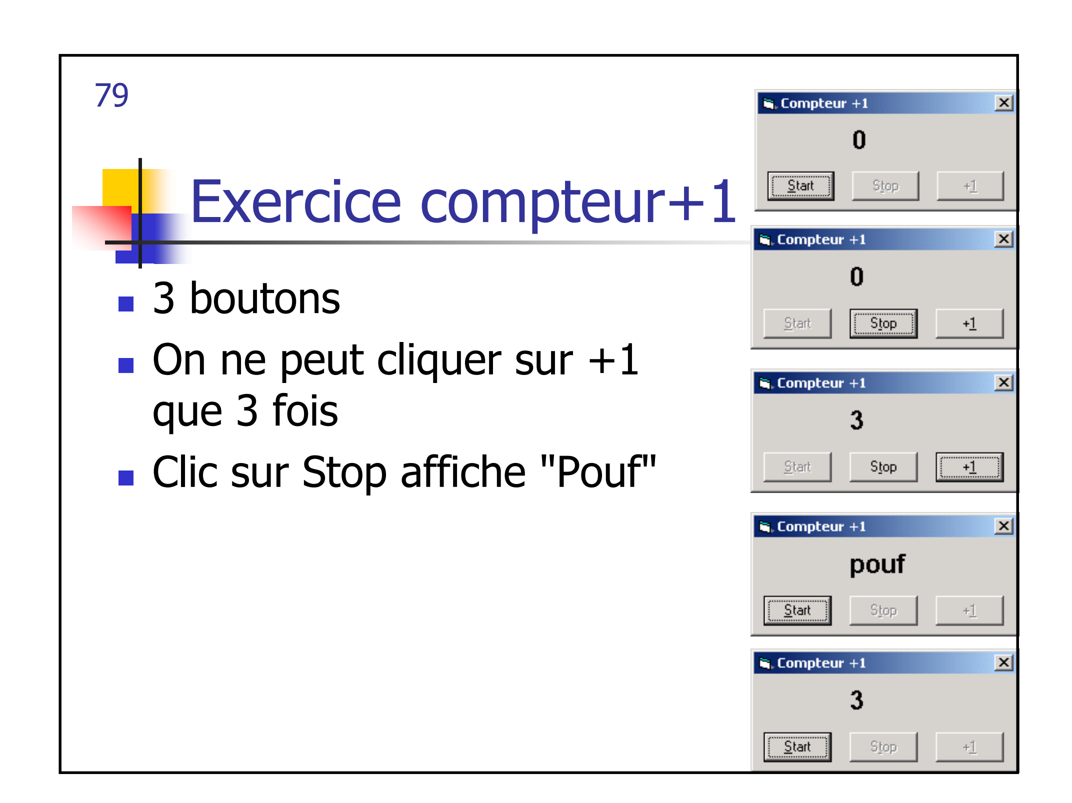
Automate Compteur Affiché

- A1:

```
For I=1 to 30000
    label.caption=I
    Label.refresh
Next I
```



- A2:
  Label.caption="Pouf"



80

Exercice compteur+1

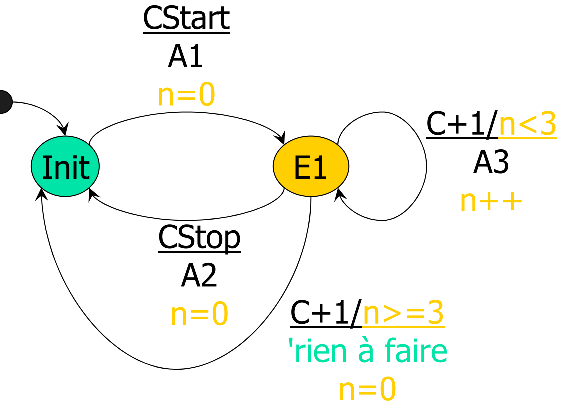
- A1: label1.caption=0
- A2: label1.caption= "pouf"
- A3: label1.caption=val(label1.caption)+1

Compteur+1 repliage

Compteur+1 repliage



Compteur+1 déterminisme



Rajout d'une variable d'état liées à l'état E1

Systématiquement mise à jour à chaque changement d'état

Ajout de conditions pour lever l'indéterminisme



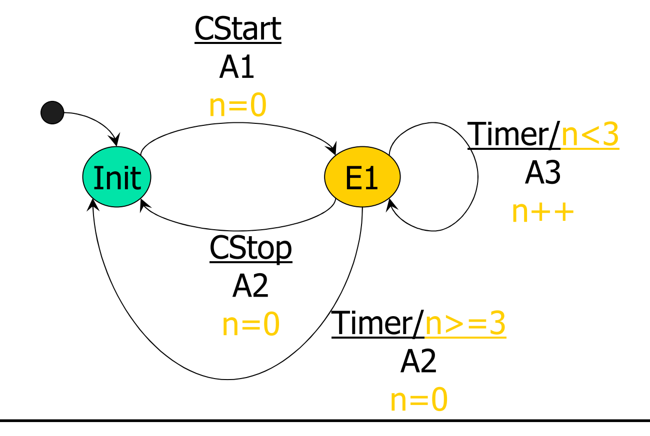
Exercice Timer (sujet)

- On enlève le bouton +1
- On rajoute un timer
  - Propriétés "interval" et "Enabled"
  - Evénement "Timer"



Exercice Timer (automate)

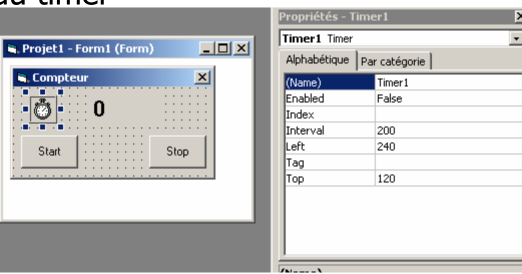
- Evénements : Cstart, Cstop, Timer



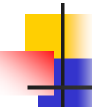


Exercice Timer (code)

- On enlève le bouton +1
- On rajoute un timer
- On donne le "name" du bouton comme "name" au timer
- On déplace le code de l'event handler de +1 vers le timer
- On definit la période du timer

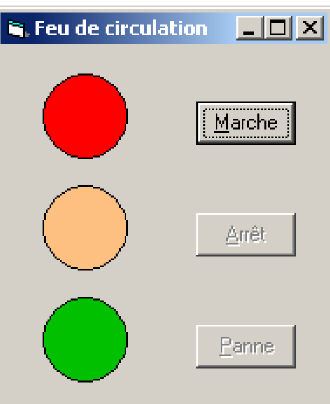






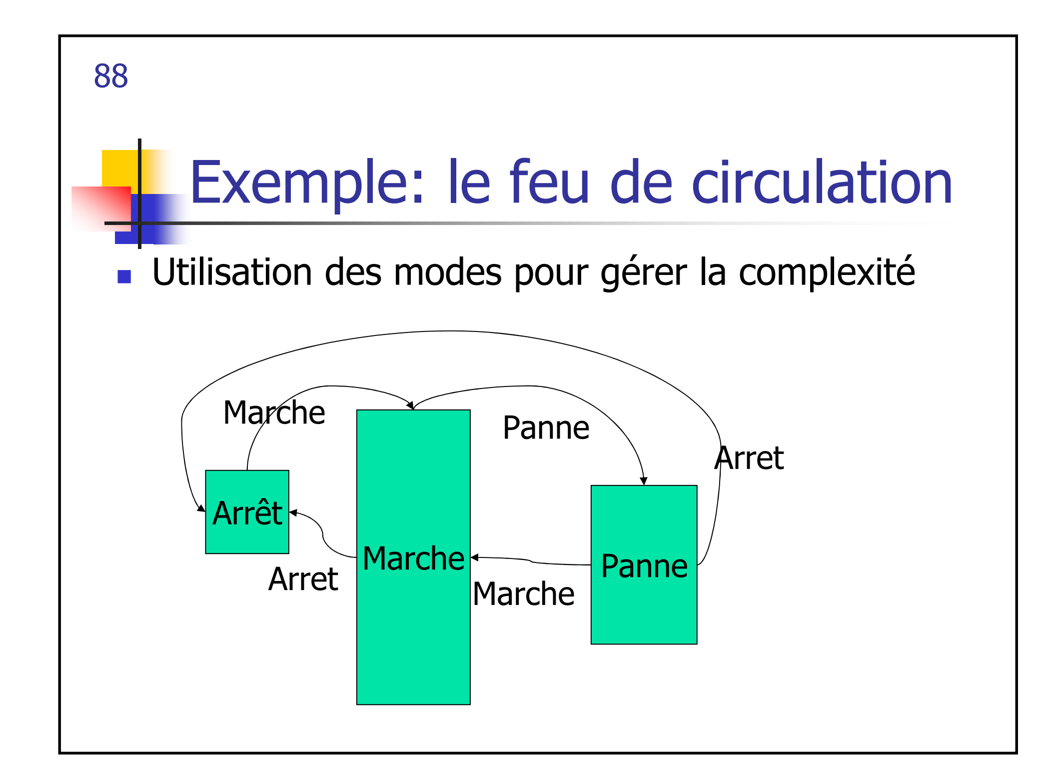
Exemple: le feu de circulation

- Spécification du comportement d'une application de gestion de feu de circulation



Exemple: le feu de circulation

- Utilisation des modes pour gérer la complexité



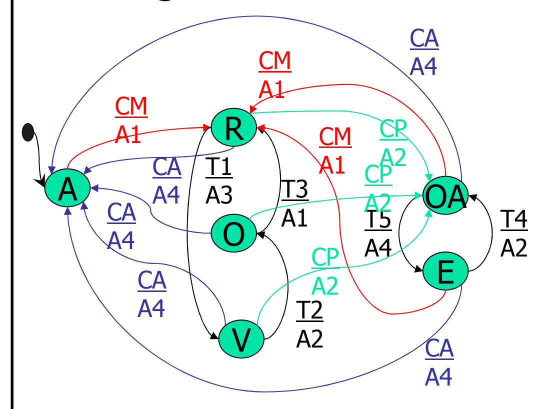


89



Exemple: le feu de circulation

- Spécification du comportement d'une application de gestion de feu de circulation



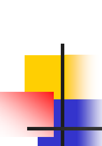
A1 : allumer Rouge
éteindre Orange
éteindre Vert

A2 : éteindre Rouge
allumer Orange
éteindre Vert

A3 : éteindre Rouge
éteindre Orange
allumer Vert

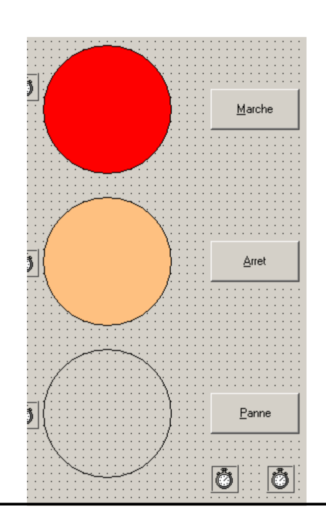
A4 : éteindre Rouge
éteindre Orange
éteindre Vert

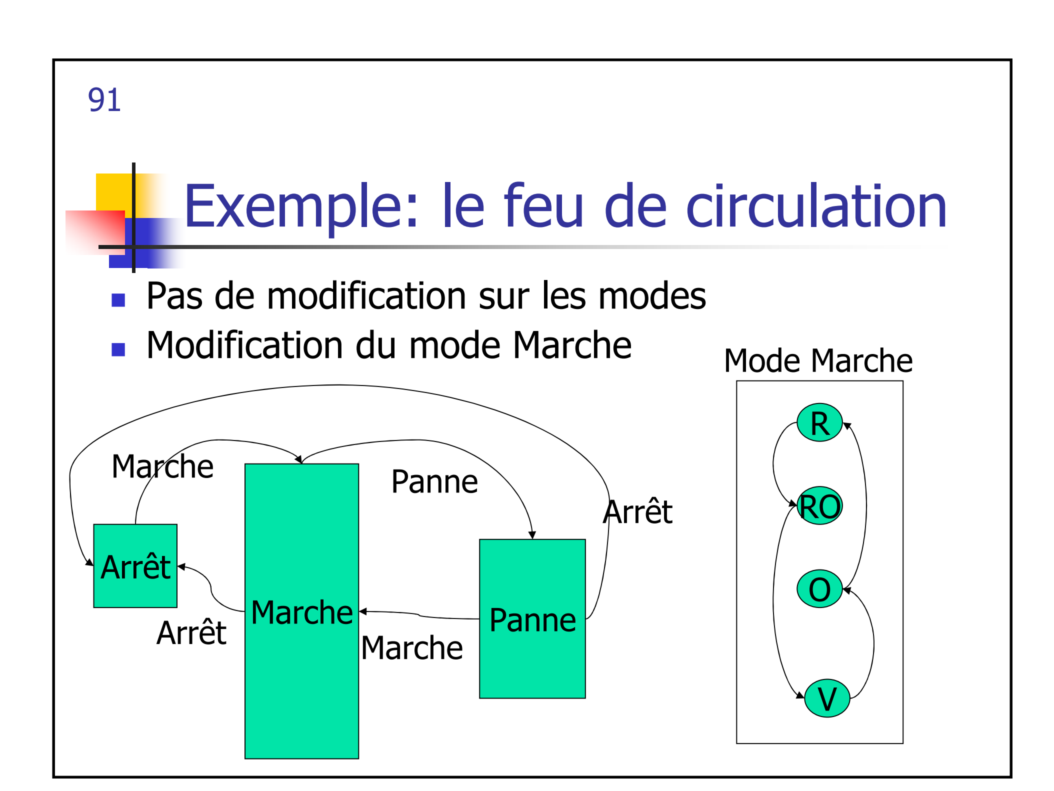
90



Exemple: le feu de circulation britanique

- Fonctionnement différent
  - Rouge et orange allumés
  - Avant le vert allumé
  - Après le rouge allumé seul





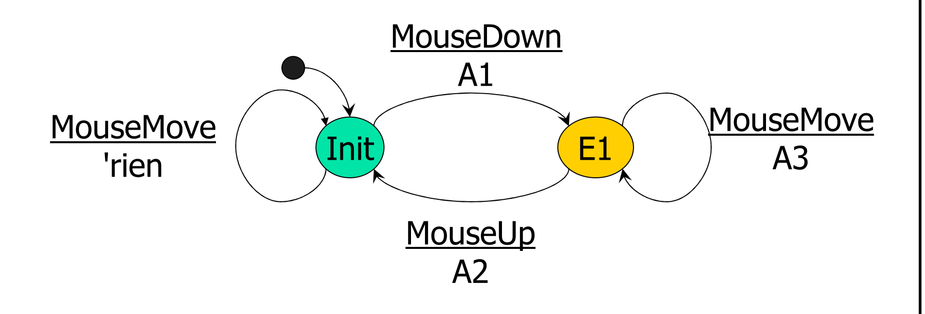


Rubber Banding

- Dessin de ligne interactif
- Bouton enfoncé commence le dessin
- Déplacement dessine
- Relâcher le bouton termine le trait



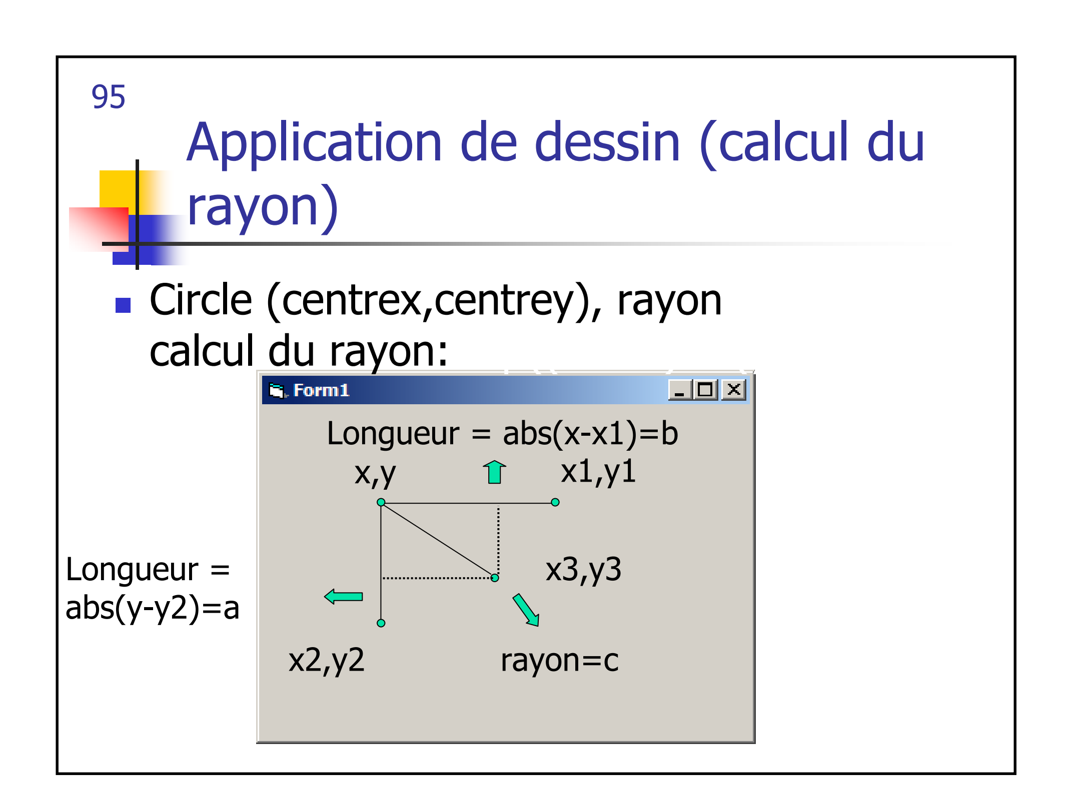
Rubber Banding





Application de dessin (sujet)

- Faire une application qui permet de dessiner des lignes, cercles, rectangles, …
- Fonctions
  - Circle (centrex,centrey), rayon
  - Line (x1,y1)-(x2,y2), RGB (0,0,0), B (trace un rectangle englobant la ligne)
- Dissocier le code de présentation des event handlers
- Gérer les modes de l'application



Application de dessin (modes)

ClicLigne

ClicCercle

ClicRectangle

Modes

Ligne

ClicLigne

Cercle

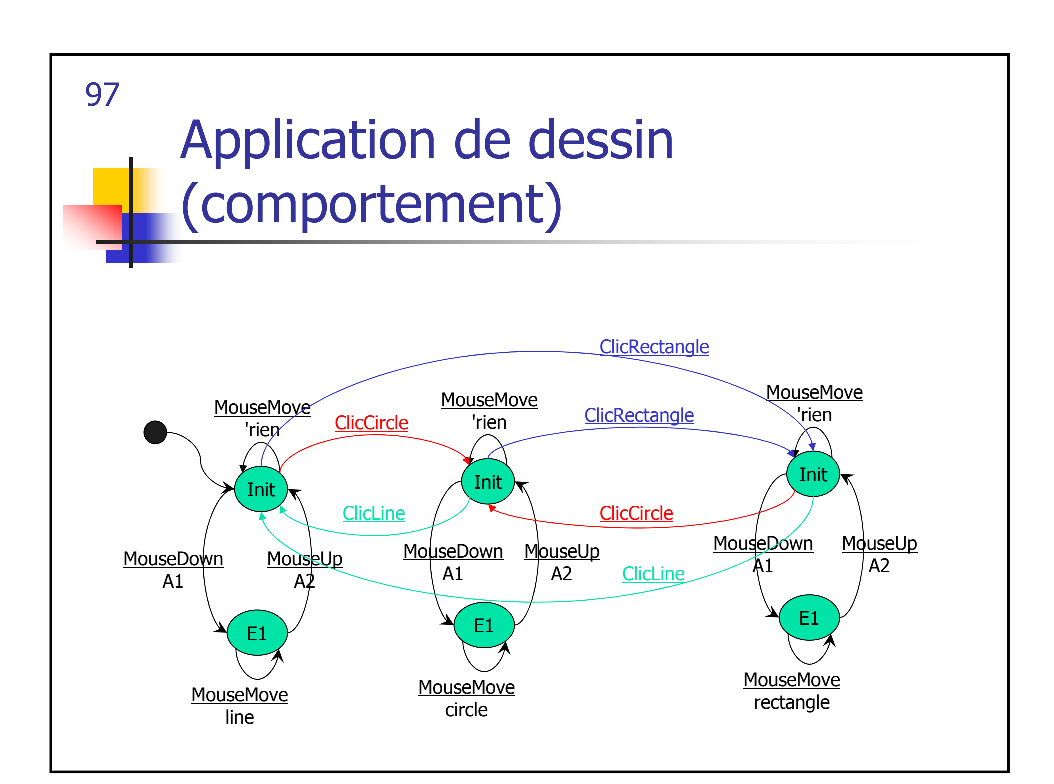
ClicCercle

Rectangle

Idle

Trace

ClicRectangle



98



Application de dessin

- Autre mise en œuvre: design pattern state as object
- VB ne permet pas l'utilisation de pointeurs de fonctions
  - On va le simuler par l'utilisation d'une variable
  - Exploiter cette variable dans la partie présentation (avec un case)
- Le comportement reste le même
- Pas d'activation et de désactivation des boutons



Polyline

- Rubber banding
- Plusieurs points
- Clic droit enlève un point
- Clic gauche rajoute un point
- Espace termine le dessin (en rouge)



Polyline (solution)

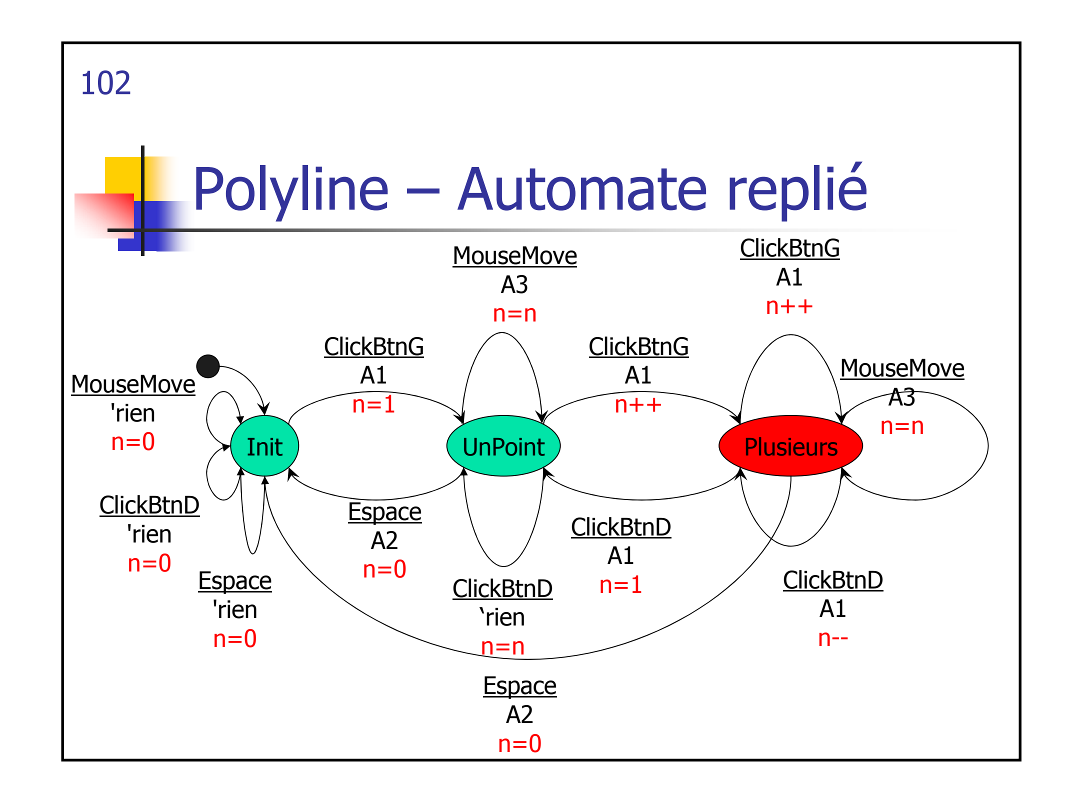
- Attention aux événements (différents des event handlers)
- Attention "clics" et pas autre chose
- Nombre d'états illimités => repliage de l'automate



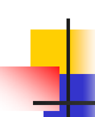
Polyline (solution)

- 4 événements
  - Clic bouton droit
  - Clic bouton gauche
  - Espace
  - Mouse_move

Polyline – Automate replié



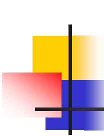
103



Application de dessin (modification)

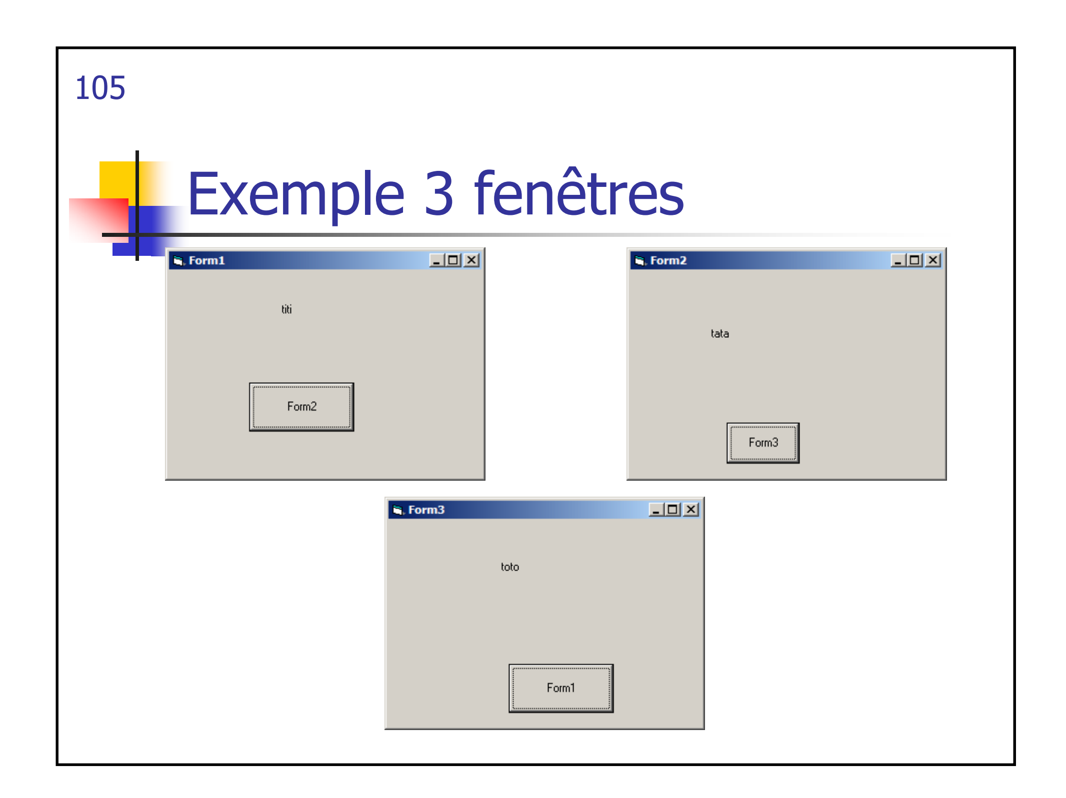
- Faire une application qui permet de dessiner des lignes, cercles, rectangles, …
- et maintenant on rajoute la polyligne …
- et on a la possibilité de modifier les couleurs
- On peut changer l'interface
  - Avoir des menus
  - Rajouter des toolbars

104



Gestion de la navigation entre fenêtres

- Form1 load: charge une fenêtre
- L'activation et la désactivation d'une fenêtre se fait par affichage (Form1.show) ou masquage (form1.hide) ou premier plan (form1.setfocus)
- Le processus de conception reste le même
  - Événements
  - Automate
  - Event handlers



106



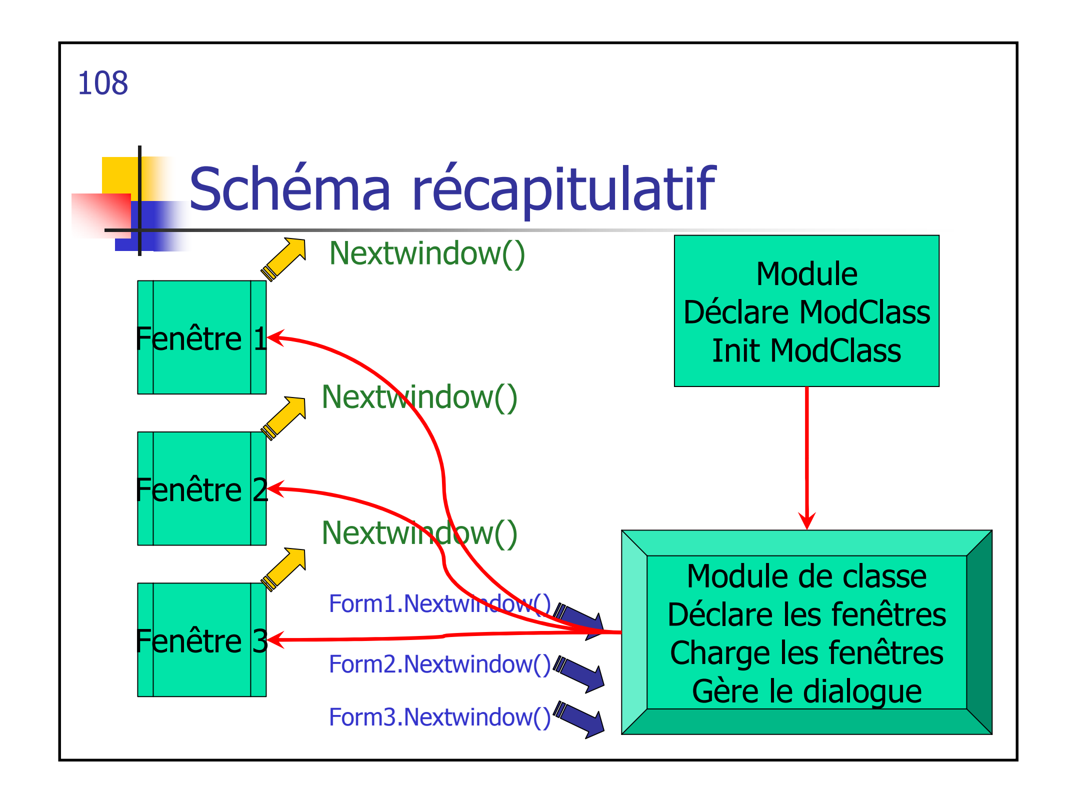
Principes VB (1/2)

- Les fenêtres vont émettre des événements
  - Elles ne peuvent pas connaître la personne en charge du dialogue entre fenêtre
  - Déclaration de l'événement dans la form (Public Event nextwindow ())
  - Utilisation de l'instruction RaiseEvent
- 1 module de classe:
  - seul moyen de faire du "catch"
  - Gère l'activation et la désactivation des fenêtres

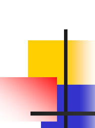


Principes VB (2/2)

- Utilisation d'un module
  - Déclare le module de classe (Public modClass As New Class1)
  - Lance l'initialisation du module de classe (on utilise une procédure de démarrage Sub Main() 1 module de classe:
- Mise en relation fenêtres/module de classe
  - Déclaration de chaque fenêtre dans le module de classe (Public WithEvents feuille1 as Form1)
  - A l'initialisation de la fenêtre affectation du nom de la fenêtre à la variable correspondante dans le module de classe   (Set modClass.feuille3 = Me)



109

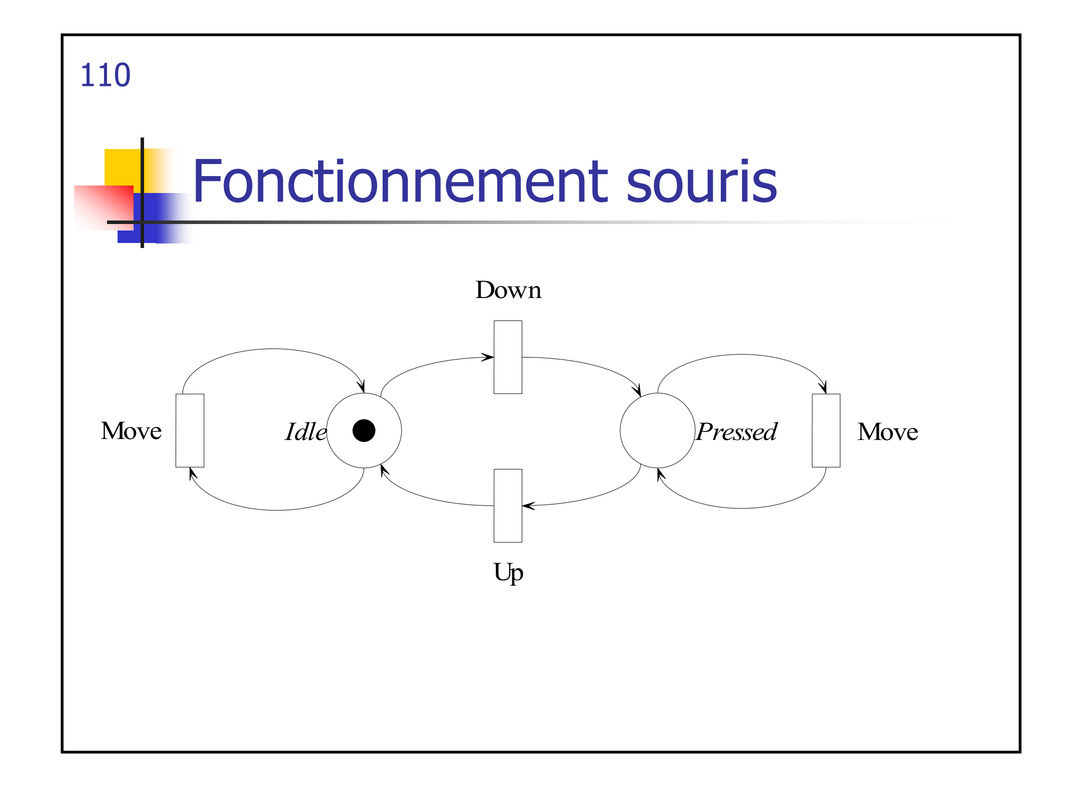


Description du comportement des transducers (drivers)

- Comportement de base d'une souris
- Génération d'événements de haut niveau
  - l'exemple du double-clic
  - Les événements drag&drop
  - Les aspects temporels

110

Fonctionnement souris



111

Le transducer souris

d

u

d

Idle

Down

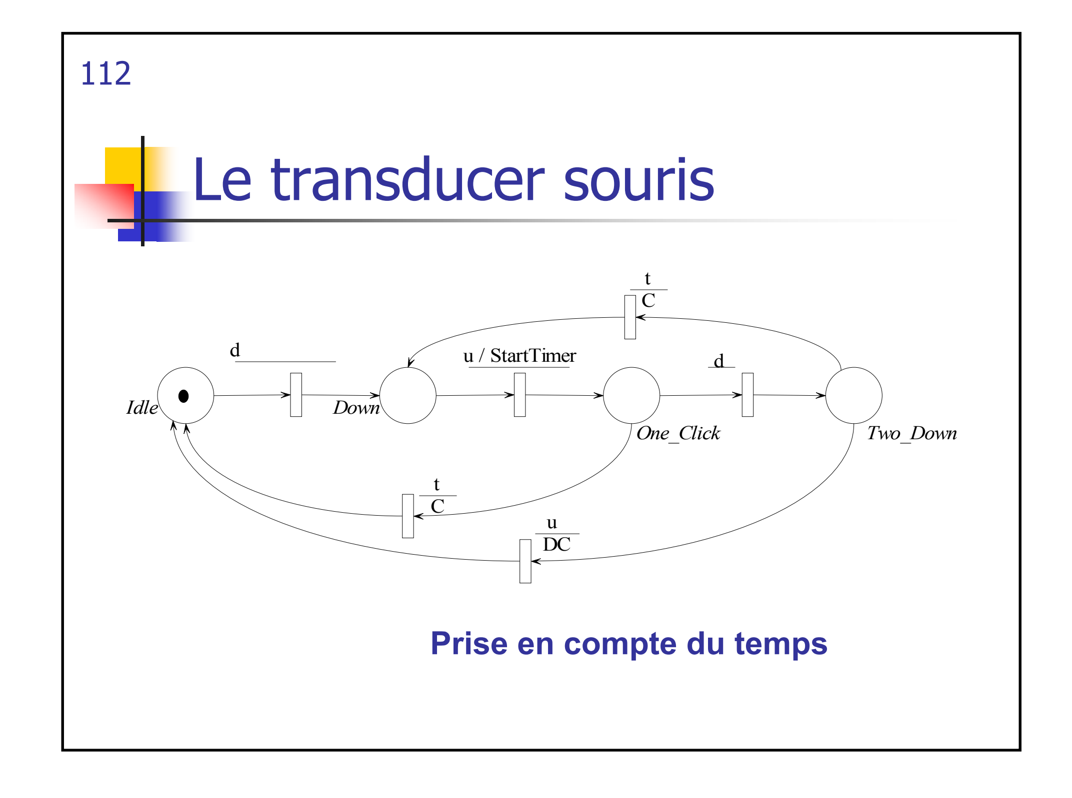
One_Click

Two_Down

u

DC

Cas simple



113

Le transducer souris

Prise en compte des mouvements

114

Plusieurs Souris ?

Multimodalité